Module 4
Selection Statements

Tamer Abdou, PhD

IND 830 - Python Programming for Data Science

Data Science Laboratory

Ryerson University

Ryerson University
The Chang School of Continuing Education

Outline

1. Comparison Expressions
2. Types of Selection Statements

Comparison Expressions

1 Comparison Expressions
The Three-valued Logic
Relational & Logical Operators

2 Types of Selection Statements

The Three-valued Logic

- There are only two boolean objects: `False` and `True`
- There is only one NoneType object: `None`
- `None` is frequently used to represent the absence of a value
- The objects `None, False, and True` cannot be reassigned

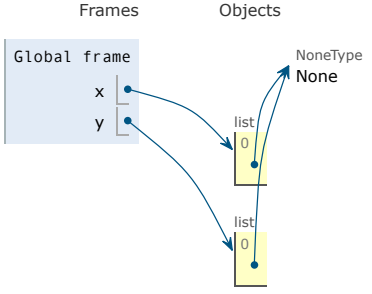
Example 4.1: Identity vs. Equality Operator

Write code in Python 3.6
(drag lower right corner to resize code editor)

```
1  x = [None]
2  y = [None]
3  print(x is y)
4  print(x == y)
```

Print output (drag lower right corner to resize)

```
False
True
```



Relational & Logical Operators

Table 4.1: Precedence rules of arithmetic and logical operators

Rank	Operator	Syntax
1	$**$	Exponentiation
2	$*, /, \%$	Multiplication, division, modulus
3	$+, -$	Addition , Subtraction
4	$==, !=, <, >, <=, >=$	Relational Operators
5	*not*	Logical Negation
6	*and*	Logical Conjunction
7	*or*	Logical disjunction

These rules come with two exceptions:

- Operations of equal precedence are **left associative**
- Exponentiation operations are **right associative**

The Logical Operator and

Table 4.2: The Logical and in Python

and	True	None	False
True	True	None	False
None	None	None	False
False	False	None	False

The Logical Operator `or`

Table 4.3: The Logical `or` in Python

or	True	None	False
True	True	True	True
None	True	None	None
False	True	False	False

Types of Selection Statements

1 Comparison Expressions

2 Types of Selection Statements
- One-way Selection Statements
- Two-way Selection Statements
- Multi-way Selection Statements

One-way Selection Statements: Syntax and Structure

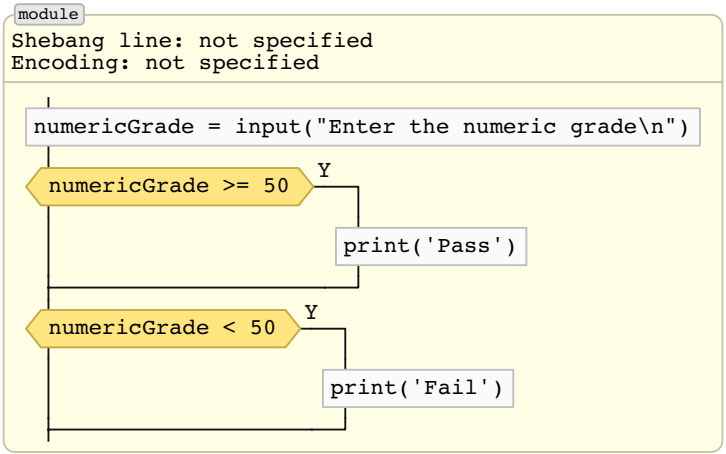
```
if <condition>:
   <sequence of statements>
```

- These statements consist of a condition and just a single sequence of statements
- The `<condition>` is a Boolean expression which consists of the Boolean values: **True** or **False**
- A Boolean value is a result of a comparison expression. These expressions includes the following relational operators: $==, !=, <, >, <=, >=$
- The `<sequence of statements>` must be indented at least one space beyond the reserved word `'if'`
- If the condition is **True**, the sequence of statements is executed. Otherwise, control proceeds to the next statement outside the selection statement

Example 4.2 (Script)

```
1 numericGrade = input("Enter the numeric grade\n")
2 if numericGrade >= 50:
3     print('Pass')
4 if numericGrade < 50:
5     print('Fail')
```

Example 4.2 (Control Flow)



Example 4.2 (Program Execution)

Write code in Python 3.6 (drag lower right corner to resize code editor)

```
1 numericGrade = input("Enter the numeric grade\n")
2 if numericGrade >= 50:
3     print('Pass')
4 if numericGrade < 50:
5     print('Fail')
```

Print output (drag lower right corner to resize)

```
Enter the numeric grade
40
```

Frames Objects

Global frame

numericGrade 40

Two-way Selection Statements: Syntax and Structure

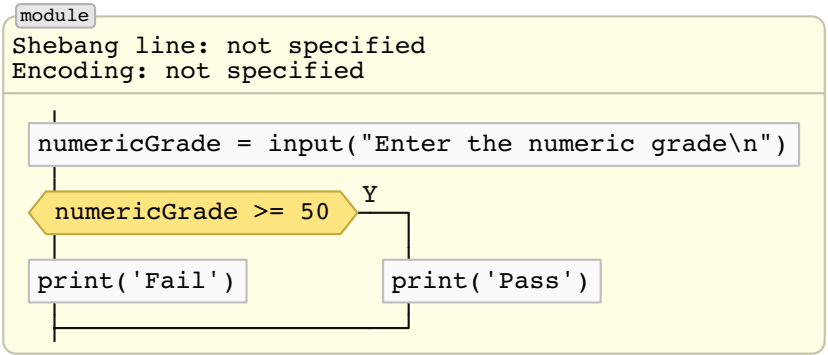
```
if <condition>:
   <sequence of statements – 1>
else:
   <sequence of statements – 2>
```

- These statements direct the compiler to make a choice between two sequences of statements
- The `<condition>` must be a Boolean expression which consists of the Boolean values: **True** or **False**
- A Boolean value is a result of a comparison expression. These expressions includes the following relational operators: $==, !=, <, >, <=, >=$
- Each `<sequence of statements>` must be indented at least one space beyond both symbols 'if' and 'else'
- If the condition is **True**, the first sequence of statements is executed. Otherwise, the second sequence of statements is executed. Then control proceeds to the next statement outside the selection statement

Example 4.3 (Script)

```
1 numericGrade = input("Enter the numeric grade\n")
2 if numericGrade >= 50:
3     print('Pass')
4 else:
5     print('Fail')
```

Example 4.3 (Control Flow)



Example 4.3 (Program Execution)

Write code in Python 3.6 (drag lower right corner to resize code editor)

```
1  numericGrade = input("Enter the numeric grade\n")
2  if numericGrade >= 50:
3      print('Pass')
4  else:
5      print('Fail')
```

Print output (drag lower right corner to resize)

```
Enter the numeric grade
40
Fail
```

Frames Objects

Global frame

numericGrade 40

Multi-way Selection Statements: Syntax and Structure

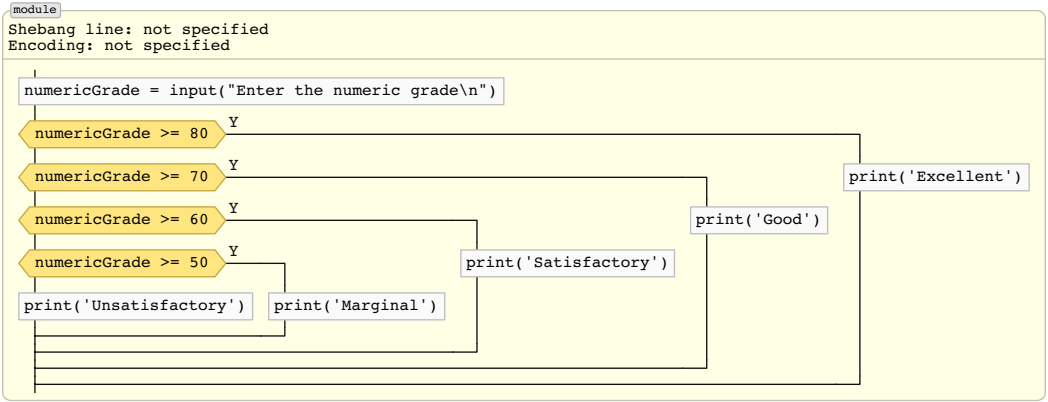
```
if <condition – 1>:
   <sequence of statements – 1>
elif <condition – 2>:
   <sequence of statements – 2>
else:
   <sequence of statements – 3>
```

- These statements direct the compiler to test several conditions and respond accordingly
- Multi-way statements consider each condition until one evaluates to **True** or they all evaluate to **False**
- When a condition evaluates to **True**, the corresponding sequence of statements is executed and control skips to the end of the entire selection statement
- If no condition evaluates to **True**, then the corresponding `<sequence of statements>` to the trailing else is executed

Example 4.4 (Script)

```
1  numericGrade = input("Enter the numeric grade\n")
2  if numericGrade >= 80:
3      print('Excellent')
4  elif numericGrade >= 70:
5      print('Good')
6  elif numericGrade >= 60:
7      print('Satisfactory')
8  elif numericGrade >= 50:
9      print('Marginal')
10 else:
11     print('Unsatisfactory')
```

Example 4.4 (Control Flow)



Example 4.4 (Program Execution)

Write code in Python 3.6 (drag lower right corner to resize code editor)

```
1  numericGrade = input("Enter the numeric grade\n")
2  if numericGrade >= 80:
3      print('Excellent')
4  elif numericGrade >= 70:
5      print('Good')
6  elif numericGrade >= 60:
7      print('Satisfactory')
8  elif numericGrade >= 50:
9      print('Marginal')
10 else:
11     print('Unsatisfactory')
```

Print output (drag lower right corner to resize)

```
Enter the numeric grade
40
Unsatisfactory
```

Frames Objects

Global frame

numericGrade 40

References

- Lambert, K. A. (2010). Control Statements. In *Fundamentals of Python: From first programs through data structures* (pp. 91-101). Boston, MA: Cengage Learning

- Python Software Foundation. (2020). Python 3.7.8 documentation. Retrieved from `https://docs.python.org/3.7/index.html`.